Parallel netCDF: News Archive

1.8.0

- **December 19, 2016**: PnetCDF **1.8.0** is released (the latest stable version). See ReleaseNotes-1.8.0.
- **November 15, 2016**: a preview release of PnetCDF **1.8.0** is available. See Changes from 1.7.0.
- **March 3, 2016**: PnetCDF **1.7.0** is released (the latest stable version). See ReleaseNotes-1.7.0.
- We continue working with the netCDF team at Unidata to improve CDF-5 and PnetCDF features in netCDF-4.

1.7.0pre1

- **January 12, 2016**: PnetCDF **1.7.0pre1** is available as a pre-release of version 1.7.0. See release note.
- We work with the netCDF team at Unidata to integrate CDF-5 and PnetCDF features in netCDF-4. See release note of 4.4.0 candidate RC4 for more information.

1.6.1

- **June 1, 2015**: PnetCDF **1.6.1** is released (the latest stable version). See ReleaseNotes-1.6.1.
- We are developing a patch to teach netCDF-4 to access CDF-5 files both in sequential and parallel. We encourage netCDF-4 users to give it a try and welcome all comments.

1.6.0

- **February 2, 2015**: PnetCDF **1.6.0** is released. See ReleaseNotes-1.6.0.

1.6.0.pre1

- **January 3, 2015**: a test release of PnetCDF **1.6.0.pre1** is available. See ReleaseNotes-1.6.0.pre1 for more details.

1.5.0

- **July 8, 2014**: PnetCDF **1.5.0** is released. See ReleaseNotes-1.5.0.
- C++ APIs are now available in 1.5.0.

1.5.0.pre1

- **May 16, 2014**: a test release of PnetCDF **1.5.0.pre1** is available. See ReleaseNotes-1.5.0.pre1 for more details.

1.4.1

- **December 23, 2013**: PnetCDF **1.4.1** is released. See ReleaseNotes-1.4.1 for more details.
- Fortran header file, pnetcdf.inc, now can be included in both fixed and free-formed Fortran programs.

- Initial _subfiling feature has been added to 1.4.1.

1.4.0

- **November 17, 2013**: PnetCDF **1.4.0** is released. See ReleaseNotes-1.4.0 for more details.
- Fortran 90 APIs are now available in 1.4.0.
- New APIs, ncmpi_get/put_varn_<type> for reading/writing a list of sub-requests to a single variable. Available for F77 and F90 as well.
- FLASH-IO benchmark using PnetCDF is now part of the source code release.

1.4.0.pre1

- **September 19 2013**: PnetCDF **1.4.0.pre1** test release. See ReleaseNotes-1.4.0.pre1 for more details.

1.3.1

- **September 24, 2012**: PnetCDF **1.3.1** released. See ReleaseNotes-1.3.1 for more details.

1.3.0

- **26 June 2012**: PnetCDF **1.3.0** released. See ReleaseNotes-1.3.0 for more details.
- In the 1.3.0 release, the unsigned and 64-bit integer data types are supported for CDF-5 format. The unsigned data types include NC_UBYTE, NC_USHORT, NC_UINT, and NC_UINT64. The 64-bit integer data types are NC_INT64 and NC_UINT64.
- New APIs for supporting more data types are added. For C, they are ncmpi_(i)put/(i)get_var*_ushort/uint/longlong/ulonglong. For Fortran, they are nfmpi_(i)put/(i)get_var*_int8.
- A new set of "buffered"-put APIs is supported in 1.3.0 release. The nonblocking iput/iget APIs require the contents of user buffers not to be changed until the wait call completed. The bput APIs use a user attached buffer to make a copy of request data, so the user buffer is free to change once the bput call returns.
- The special character set, "special2", and multi-byte UTF-8 encoded characters introduced in the CDF-2 file format for variable, dimension, and attribute name strings are now supported.
- A set of example programs and QuickTutorial are now available.

1.2.0

- **19 August 2010**: PnetCDF **1.2.0** released. See ReleaseNotes-1.2.0 for more details.
- Nonblocking I/O is redesigned in the 1.2.0 release. It defers the I/O requests until "wait" call, so small requests can be aggregated into large ones for better performance.
- Two new hints, `nc_header_align_size` and `nc_var_align_size`, are added. The former allows pre-allocation of a larger header size to accommodate new header data in case new variables or attributed are added later. The latter aligns the starting file offsets of non-record variables. Refer to VariableAlignment for a more detailed description.
- Data consistency control has been revised. A more strict consistency can be enforced by using `NC_SHARE` mode at the file open/create time. In this mode, the file header is synchronized to the file if its contents have changed. Such file synchronization of calling `MPI_File_sync()` happens in many places, including `ncmpi_enddef()`, `ncmpi_redef()`, all APIs that change global or variable attributes,

dimensions, and number of records.

- As calling `MPI_File_sync()` is very expensive on many file systems, users can choose more relaxed data consistency, i.e. by not using `NC_SHARE`. In this case, file header is synchronized among all processes in memories. No `MPI_File_sync()` will be called if header contents have changed. `MPI_File_sync()` will only be called when switching data mode, i.e `ncmpi_begin_indep_data()` and `ncmpi_end_indep_data()`.